02 – Language modelling

IA161 Advanced Techniques of Natural Language Processing

P. Rychlý, V. Baisa

NLP Centre, FI MU, Brno

September 22, 2021

Language models—what are they good for?

- assigning scores to sequencies of words
- predicting words
- generating text

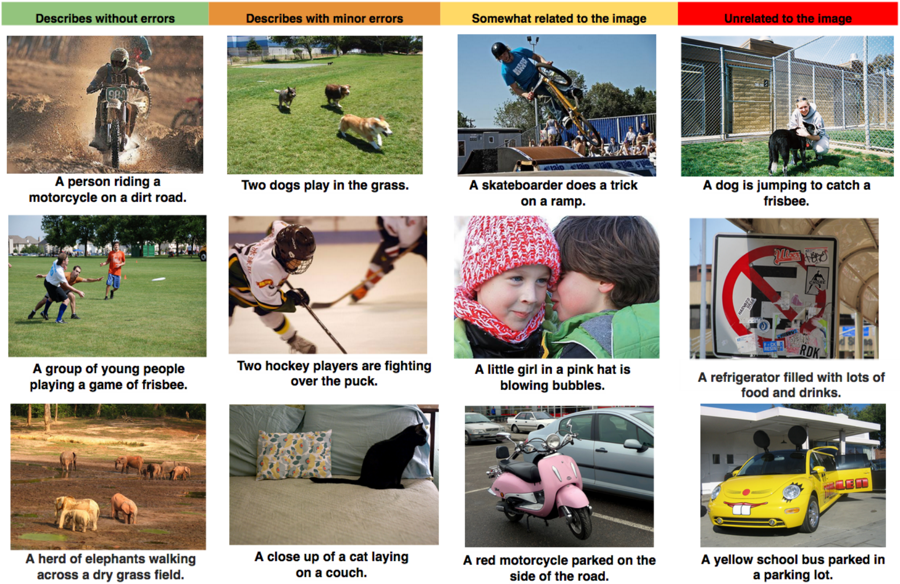
$\Rightarrow$

- statistical machine translation
- automatic speech recognition
- optical character recognition

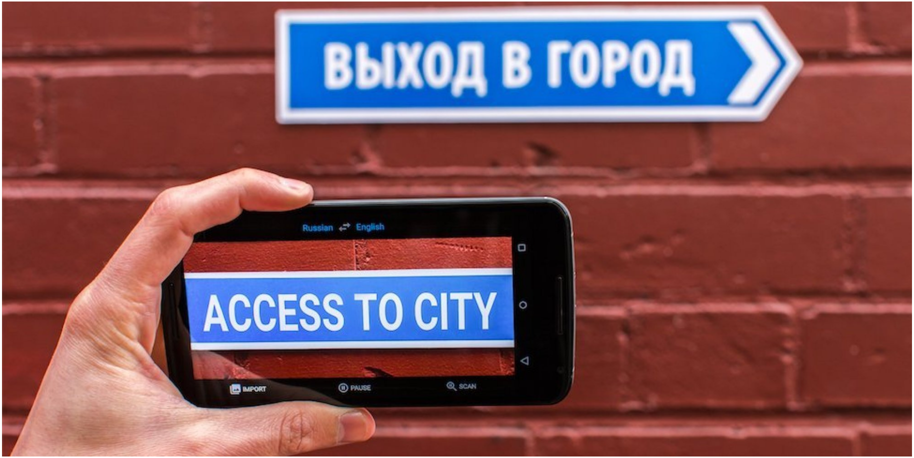
Predicting words

Do you speak ...
Would you be so ...
Statistical machine ...
Faculty of Informatics, Masaryk ...
WWII has ended in ...
In the town where I was ...
Lord of the ...

Generating text



MT + OCR



ВЫХОД В ГОРОД
Russian English
ACCESS TO CITY
IMPORT
PAUSE
SCAN

OCR

Raw Article Scan

Road hauliers are seeking, and in many cases obtaining, increases in rates ranging from 3½ per cent. to 6 per cent.

This emerged yesterday from an area-by?area survey carried out by THE FINANCIAL TIMES, a fortnight after publication of the report of the National Board for Prices and Incomes on the road haulage industry.

Hauliers claimed that the report was having the effect of plolonging negotiations, but said they were confident that eventually they would win rises of the size they originally contemplated.

Meanwhile, representatives of the Road Haulage Association may discuss aspects of the N.B.P.I. report with union officials in London to-day at the inaugural meeting of the industry's new 24-strong negotiating committee.

This body, which was established some weeks ago, is the one on

Financial Times 13th July 1965, Page One

3rd Party OCR

ROaR""^'*
i
1.
H
24Wy"
m..
m
.!;
*4
-,ta~
a
·*P
sr;i
r·~·
i
.l~s
24
a·
·

Tesseract OCR
(with default settings)

Road hauliers are seeking, and
in many cases obtunin.
increasesin rates ranging from 3A per cent.
to 6 per cent.
This emerged yesterday from an
area~b ?area survey carried out by
Tue mmcuu. Tums, a fortnight
after publication of the report of
the National Board for Prices and
incomes on the med haulage in-
dustry.
Heuliers claimed that the report
was having the e\ufb01ect of plolong-
ing negotiations. but said they
were con\ufb01dent that eventually they
would win rise: of the tile they
originally contemplated.
Meanwhile. representative: of
the Road Haulage Associntion may
disease aspects of the N.B.P.l. re-
port wrth union of\ufb01ciele in London
to-dey at the inaugural meettng of \u2018
thenndustry\u2018s. new 244trong nego-
: tinting committee. ,
This body. which we! established
some weeks ago. in the one on

Language models – probability of a sentence

- LM is a probability distribution over all possible word sequences.
- What is the probability of utterance of s?

Probability of sentence

p_{LM}(Catalonia President urges protests)
p_{LM}(President Catalonia urges protests)
p_{LM}(urges Catalonia protests President)
...

Ideally, the probability should strongly correlate with fluency and intelligibility of a word sequence.

N-gram models

- an approximation of long sequencies using short n-grams
- a straightforward implementation
- an intuitive approach
- good local fluency

Randomly generated text

"Jsi nebylo vidět vteřin přestal po schodech se dal do deníku a položili se táhl ji viděl na konci místnosti 101," řekl důstojník.

Hungarian

A társaság kötelezettségeiért kapta a középkori temploma az volt, hogy a felhasználók az adottságai, a felhasználó azonosítása az egyesület alapszabályát.

N-gram models, naïve approach

$$W = w_1, w_2, \cdots, w_n$$

$$p(W) = \prod_i p(w_i | w_1 \cdots w_{i-1})$$

Markov's assumption

$$p(W) = \prod_i p(w_i | w_{i-2}, w_{i-1})$$

$$p(\textit{this is a sentence}) = p(\textit{this}) \times p(\textit{is} | \textit{this}) \times p(a | \textit{this}, \textit{is}) \times p(\textit{sentence} | \textit{is}, a)$$

$$p(a | \textit{this}, \textit{is}) = \frac{|\textit{this is a}|}{|\textit{this is}|}$$

Sparse data problem.

Computing, LM probabilities estimation

Trigram model uses 2 preceding words for probability learning. Using **maximum-likelihood estimation**:

$$p(w_3|w_1, w_2) = \frac{count(w_1, w_2, w_3)}{\sum_w count(w_1, w_2, w)}$$

quadrigram: *(lord, of, the, ?)* ()

w	**count**	$p(w)$
rings	30,156	0.425
flies	2,977	0.042
well	1,536	0.021
manor	907	0.012
dance	767	0.010
...		

Large LM – n-gram counts

How many unique n-grams in a corpus?

order	unique	singletons
unigram	86,700	33,447 (38.6%)
bigram	1,948,935	1,132,844 (58.1%)
trigram	8,092,798	6,022,286 (74.4%)
4-gram	15,303,847	13,081,621 (85.5%)
5-gram	19,882,175	18,324,577 (92.2%)

Corpus: Europarl, 30 M tokens.

Language models smoothing

The problem: an n-gram is missing in the data but is in a *sentence* $\rightarrow$ $p(sentence) = 0$.

We need to assign non-zero p for *unseen data*. This must hold:

$$\forall w . p(w) > 0$$

The issue is more pronounced for higher-order models.

Smoothing: an attempt to amend real counts of n-grams to expected counts in any (unseen) data.

Add-one, Add-α, Good–Turing smoothing

Deleted estimation

We can find unseen n-grams in another corpus. N-grams contained in one of them and not in the other help us to estimate general amount of unseen n-grams.

E.g. bigrams not occurring in a training corpus but present in the other corpus million times (given the amount of all possible bigrams equals 7.5 billions) will occur approx.

$$\frac{10^6}{7.5 \times 10^9} = 0.00013\times$$

Interpolation and back-off

Previous methods treated all unseen n-grams the same. Consider trigrams

beautiful young girl
beautiful young granny

Despite we don't have any of these in our training data, the former trigram should be more probable.

We will use probability of lower order models, for which we have necessary data:

young girl
young granny
beautiful young

Interpolation

$$p_I(w_3|w_1w_2) = \lambda_1 p(w_3) \times \lambda_2 p(w_3|w_2) \times \lambda_3 p(w_3|w_1w_2)$$

If we have enough data we can trust higher order models more and assign a higher significance to corresponding n-grams.

p_I is probability distribution, thus this must hold:

$$\forall \lambda_n : 0 \leq \lambda_n \leq 1$$

$$\sum_n \lambda_n = 1$$

Quality and comparison of LMs

We need to compare quality of various LM (various orders, various data, smoothing techniques etc.)

1) extrinsic (WER, MT, ASR, OCR) and 2) intrinsic (perplexity) evaluation

A good LM should assign a higher probability to a good (looking) text than to an incorrect text. For a fixed test text we can compare various LMs.

Cross-entropy

$$
\begin{aligned}
H(p_{LM}) &= -\frac{1}{n} \log p_{LM}(w_1, w_2, \dots w_n) \\
&= -\frac{1}{n} \sum_{i=1}^{n} \log p_{LM}(w_i | w_1, \dots w_{i-1})
\end{aligned}
$$

Cross-entropy is average value of negative logarithms of words probabilities in testing text. It corresponds to a measure of uncertainty of a probability distribution. **The lower the better**.

A good LM should reach entropy close to real entropy of language. That can't be measured directly but quite reliable estimates exist, e.g. Shannon's game. For English, entropy is estimated to approx. 1.3 bit per letter.

Perplexity

$$PP = 2^{H(p_{LM})}$$

Perplexity is a simple transformation of cross-entropy.

A good LM should not waste p for improbable phenomena.

The lower entropy, the better $\rightarrow$ the lower perplexity, the better.

Comparing smoothing methods (Europarl)

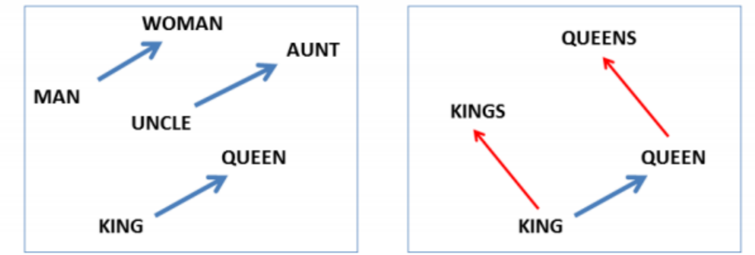
method	**perplexity**
add-one	382.2
add-α	113.2
deleted est.	113.4
Good–Turing	112.9

Neural Networks

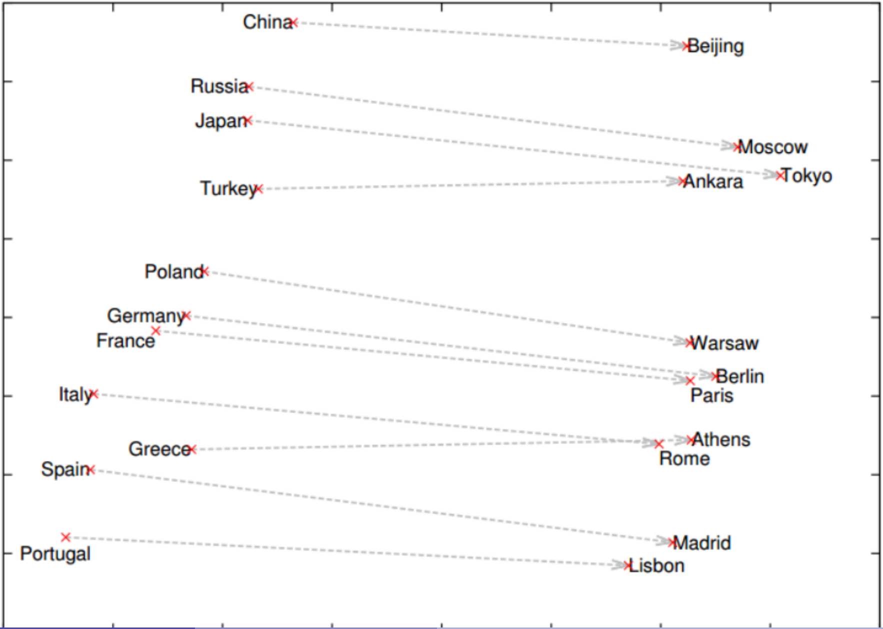
- no probabilities, only scores
- One-hot representation of words: `[ 0 0 0 0 0 0 1 0 0 0 0 ]`
- adapting a model means changes in the whole network

Distributional Representation of Words

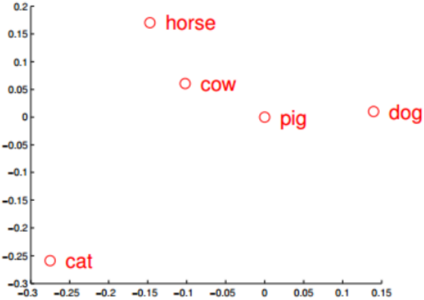
- goal: more compact representation of vectors
- limited dimensionality (500–1000)
- [Mikolov et al., 2013]
- word vectors capture many linguistic properties (gender, tense, plurality, even semantic concepts like “capital city of”)

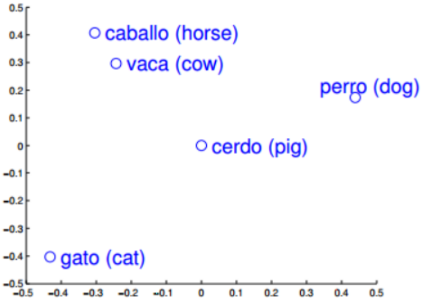


Features: vector arithmetics I



Features: vector arithmetics II

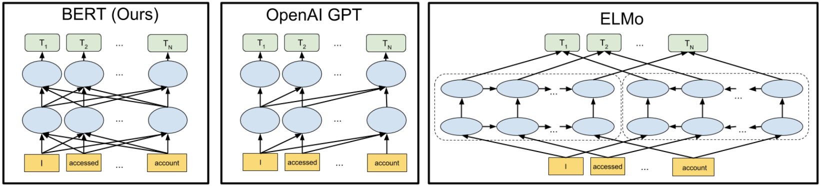




State-of-the-art neural models

Using context to compute token/sentence/document embedding via transformes.[Vaswani et al., 2017]

- BERT = Bidirectional Encoder Representations from Transformers [Devlin et al., 2018]
- GPT = Generative Pre-trained Transformer [Brown et al., 2020]
- many varians: tokenization, attention, encoder/decoder connections



BERT

- Google
- pre-training on raw text
- masking tokens, is-next-sentence
- big pre-trained models available
- domain (task) adaptation

Input: The man went to the [MASK]$_1$. He bought a [MASK]$_2$ of milk .
Labels: [MASK]$_1$ = store; [MASK]$_2$ = gallon

Sentence A = The man went to the store.
Sentence B = He bought a gallon of milk.
Label = IsNextSentence

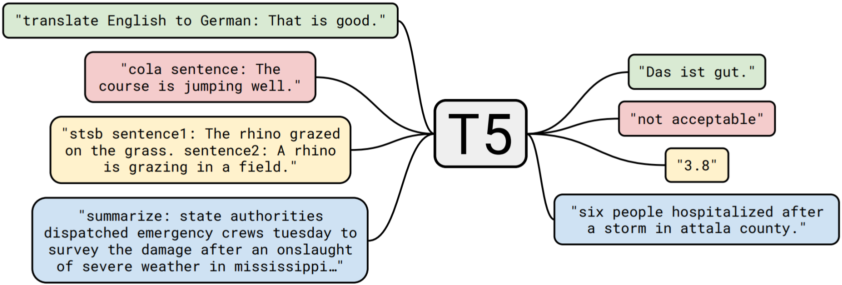
Sentence A = The man went to the store.
Sentence B = Penguins are flightless.
Label = NotNextSentence

GPT

- Open AI
- GPT-2: 1.5 billion parameters
- GPT-3: 175 billion parameters
- very good text generation
  $\rightarrow$ potentially harmful applications
- Misuse of Language Models
- bias – generate stereotyped or prejudiced content:
  gender, race, religion
- Sep 2020: Microsoft have "exclusive" use of GPT-3

T5: Text-To-Text Transfer Transformer

- Google AI
- transfer learning
- C4: Colossal Clean Crawled Corpus



Evaluation of language models

- standard multi-task benchmarks
- GLUE (`https://gluebenchmark.com`)
- SuperGLUE (`https://super.gluebenchmark.com`)
- XTREME Cross-Lingual Transfer Evaluation of Multilingual Encoders (`https://sites.research.google/xtreme`)
- perplexity is not used anymore

Libraries and Frameworks

- Dive into Deep Learning: online book
  `https://d2l.ai`
- Hugging Face Transformers: many ready to use models
  `https://huggingface.co/transformers`
- jiant: library, many tasks for evaluation
  `https://jiant.info`
- GluonNLP: reproduction of latest research results
  `https://nlp.gluon.ai`
- low level libraries: NumPy, PyTorch, TensorFlow, MXNet

References I

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020).
Language models are few-shot learners.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2018).
BERT: pre-training of deep bidirectional transformers for language understanding.
CoRR, abs/1810.04805.

Mikolov, T., Yih, W.-t., and Zweig, G. (2013).
Linguistic regularities in continuous space word representations.
In *HLT-NAACL*, pages 746–751.

References II

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017).
Attention is all you need.
CoRR, abs/1706.03762.

Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. (2014).
Show and Tell: A Neural Image Caption Generator.
ArXiv e-prints.

Low-level language models

The goal of the task is:

- create a simple language models from own data
  - statistical trigram LM
  - neural LM using previous/following word embeddings
- generate random text using models
- use LM in a real application – diacritics restoration

Low-level language models

Start with the colab notebook: `https://colab.research.google.com/drive/1Xhf6i-G3B4nnhn2eSNlg0QcCdLOQwjqH?usp=sharing`

- Small text for setup: *1984 book* from Project Gutenberg `https://gutenberg.net.au/ebooks01/0100021.txt`
- Czech model of Czech Europarl data from one of:
  `https://corpora.fi.muni.cz/ces-1m.txt` (1 MB)
  `https://corpora.fi.muni.cz/ces-10m.txt` (10 MB)

  `https://corpora.fi.muni.cz/ces-150m.txt` (150 MB)
- choose one task:
  - use trigram LM for diacritics restoration:
    write a function with text without diacritics as input and same text with added diacritics as a return value
  - generate text using neural LM:
    write a function to generate random text using neural LM, optional parameter is the start of the text